AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions of claims in the application:

Listing of Claims:

1. (Currently Amended) A computer implemented method for creating a language-neutral representation of a compile unit transformable to at least one of a plurality of different types of code representations, the language-neutral representation comprising:

a hierarchal arrangement of program elements that ~~agnostically~~ neutrally characterize the compile unit; and

at least one of the program elements representing a type declaration that characterizes at least one class of programmatic constructs of the compile unit.

2. (Original) The language-neutral representation of claim 1, further comprising a collection of at least one member that characterizes programmatic attributes associated with and able to be implemented within the at least one class.

3. (Original) The language-neutral representation of claim 2, wherein the collection further comprises an expression class within the at least one class.

4. (Original) The language-neutral representation of claim 2, wherein the collection further comprises a statement class within the at least one class.

5. (Original) The language-neutral representation of claim 2, wherein the hierarchal arrangement further comprises a namespace that contains the at least one class.

6. (Original) The language-neutral representation of claim 1, wherein at least one of the program elements of the hierarchal arrangement encapsulates another of the program elements.

7. (Original) The language-neutral representation of claim 1 in combination with an interface associated with the language-neutral representation, the interface being operative to enable transformation of the language-neutral representation to a corresponding desired code representation.

8. (Original) The combination of claim 7, wherein the program elements comprise objects, each object exposing at least one of a method, attribute, and property of each respective object, the interface being operative to employ the at least one of method, attribute and property to facilitate the transformation into the desired code representation.

9. (Original) The combination of claim 7, wherein the interface further comprises a compiler interface programmed to enable transformation of the language-neutral representation to a corresponding low-level language code representation.

10. (Original) The combination of claim 9, wherein the low-level language representation further comprises an assembly of computer-executable instructions.

11. (Original) The combination of claim 7, wherein the interface further comprises a code generator interface programmed to enable conversion of the language-neutral representation to a corresponding high-level language code representation.

12. (Original) The language-neutral representation of claim 1, wherein the program elements comprise instances of a plurality of language-neutral classes, each instance defining an associated object.

13. (Original) The language-neutral representation of claim 12, wherein at least one associated object represents the type declaration, at least another object being encapsulated within the at least one object representing the at least one type declaration, the at least another object representing program code of the compile unit that derives from a class associated with the at least type declaration.

14. (Currently Amended) A computer implemented method for creating a language-neutral representation of a compile unit ~~programmatic elements~~, comprising:

an instance of at least one of a plurality of language-neutral classes, the plurality of classes representing different programmatic constructs of a compile unit and having a hierarchal relationship relative to each other, whereby transformation of the instance into a different representation of the respective programmatic construct is facilitated.

15. (Original) The language-neutral representation of claim 14, further comprising a plurality of instances of the plurality of the classes, wherein each instance of a corresponding class of the plurality of classes represents a respective programmatic construct of the compile unit, the plurality of instances being organized in a hierarchal relationship based on the classes associated with the plurality of instances and relationships among the programmatic constructs represented thereby.

16. (Original) The language-neutral representation of claim 15, wherein each of the plurality of instances exposes at least one item associated with the programmatic construct represented thereby.

17. (Original) The language-neutral representation of claim 16, wherein at least one of the plurality of instances represents a type declaration, at least another instance being encapsulated within the instance representing the type declaration, the at least another instance representing a programmatic construct that derives from the at least type declaration.

18. (Original) The language-neutral representation of claim 17 wherein the at least another object further comprises at least one of a statement and an expression.

19. (Original) The language-neutral representation of claim 16 in combination with an interface that enables transformation of the representation to the different representation, the

interface being operative to employ the at least one item to facilitate the transformation of the language-neutral representation into the different representation.

20. (Original) The combination of claim 19, wherein the interface further comprises a compiler interface programmed to enable transformation of the language-neutral representation to a corresponding low-level language code representation.

21. (Original) The combination of claim 20, wherein the low-level language code representation further comprises an assembly of computer-executable instructions.

22. (Original) The combination of claim 19, wherein the interface further comprises a code generator interface programmed to enable generation of a high-level language code representation from the language-neutral representation.

23. (Currently Amended) A computer implemented method for creating a language-neutral representation of ~~computer-executable instructions~~ a compile unit that ~~are~~ is transformable to at least one other type of software code representation, the language-neutral representation comprising:

a hierarchal arrangement of objects, each object representing a different program element of the compile unit;

at least one class object that represents at least one defined class of program elements of the compile unit; and

at least one member object associated with the at least one class object that represents computer-executable instructions operable on at least some program elements in the at least one defined class.

24. (Original) The language-neutral representation of claim 23, further comprising a namespace object that represents a namespace of the compile unit, the namespace object comprising a collection of class objects including the at least one class object.

25. (Original) The language-neutral representation of claim 24, further comprising a plurality of member objects associated with the at least one class object, wherein the at least one class object represents a common base class that is shared by the plurality of member objects.

26. (Original) The language-neutral representation of claim 24, wherein the plurality of member objects further comprise a collection of objects representing at least one of a statement and an expression of the compile unit.

27. (Original) The language-neutral representation of claim 23 in combination with an associated interface, the interface being operative to enable transformation of the language-neutral representation to a corresponding desired code representation.

28. (Original) The combination of claim 27, wherein each object exposes at least one item indicative of the program element represented thereby, the interface being operative to employ each exposed item to facilitate transformation of the language-neutral representation into the desired code representation.

29. (Original) The combination of claim 28, wherein the interface further comprises a compiler interface that exposes computer-executable instructions to transform the language-neutral representation to a corresponding low-level language code representation.

30. (Original) The combination of claim 29, wherein the low-level language representation further comprises an assembly of computer-executable instructions.

31. (Original) The combination of claim 28, wherein the interface further comprises a code generator interface that exposes computer-executable instructions to convert the language-neutral representation to a corresponding high-level language code representation.

32. (Currently Amended) A computer implemented method for creating a language-neutral representation of programmatic elements, comprising:

means for representing different code portions of a compile unit in a language-neutral manner, the means for representing being arranged according to a hierarchy; and

means associated with the means for representing for exposing information about each of the means for representing to facilitate transformation of the means for representing to a desired alternative form.

33. (Original) The language-neutral representation of claim 32, further comprising means for transforming the language-neutral representation to a corresponding desired representation of code.

34. (Original) The combination of claim 33, wherein the means for transforming further comprises means for compiling the language-neutral representation to a corresponding low-level language code representation.

35. (Original) The combination of claim 33, wherein the means for transforming further comprises means for generating a corresponding high-level language code representation from the language-neutral representation.

36. (Original) A system operable to transform a language-specific representation of a compile unit to a language-neutral representation thereof, the system comprising:

a plurality of language-neutral classes that represent different programmatic constructs and have a hierarchal relationship relative to each other;

an interface that exposes the plurality of classes; and

a design system operative to employ the interface to instantiate selected classes of the plurality of classes to create corresponding objects that represent respective code elements of the compile unit arranged according to the hierarchal relationship to define the language-neutral representation.

37. (Original) system of claim 36, wherein at least one of the objects is a namespace object that comprises a collection of at least one base class member object that provides a common base class for at least some of the objects.

38. (Original) The system of claim 37, wherein the at least one base class further comprises at least one member comprising a collection of objects representing at least one of a statement and an expression of the compile unit.

39. (Original) The system of claim 36, further comprising a code generator interface programmed to transform the language-neutral representation to a high-level language code representation that is different from the language-specific representation.

40. (Original) A computer-readable medium having stored thereon computer-executable instructions for creating a language-neutral representation of a compile unit, the language-neutral representation comprising associated objects that represent different parts of the compile unit arranged according to an established hierarchy, each of the objects being an instance of a respective class of a plurality of language-neutral classes, each respective class defining a different part of code according to the established hierarchy.

41. (Original) A method to transform a language-specific representation of a compile unit to a language-neutral representation thereof, the method comprising:

mapping each of a plurality of programmatic constructs associated with the language-specific representation to a corresponding class of a plurality of language-neutral classes, the classes having a hierarchal relationship relative to each other;

instantiating each corresponding class based on the mapping to create corresponding objects that represent respective programmatic constructs of the compile unit; and

arranging the objects according to the hierarchal relationship to define the language neutral representation.

42. (Original) The method of claim 41, further comprising transforming the language-neutral representation into a corresponding high-level language code representation that is different from the language-specific representation.

43. (Original) A system to transform a language-neutral representation of a compile unit to a high-level language, comprising:

an interface operative to control manipulation of a plurality of language-neutral program elements of the compile unit, each of the plurality of program elements being an instance of a corresponding class of a plurality of classes arranged according to a hierarchy; and

a code generator operative to implement the interface to transform each of the program elements to a high-level representation thereof corresponding to program information associated with each respective program element.

44. (Original) The system of claim 43, wherein the interface further comprises a class interface to expose program information relating to a base class of the compile unit representing at least one type declaration.

45. (Original) The system of claim 44, wherein the interface further comprises a member interface operative to expose program information relating to a member of the base class of the compile unit.

46. (Original) The system of claim 43, wherein the member of the base class represents a base class for at least one of a statement and an expression.

47. (Original) A method to transform a language-neutral representation of a compile unit to a corresponding target language-based representation, the language-neutral representation including at least one language-neutral program element, the at least one program element being an instance of a corresponding class of a plurality of classes arranged according to a hierarchy, the method comprising:

providing an interface to expose at least one method operative to control at least one of manipulation of program elements and compilation of the language neutral representation; and

depending on the target language-based representation, converting the at least one program element to a high-level language-based representation thereof according to the at least one exposed method or implementing the interface relative to the language neutral representation to compile each instance of the language-neutral representation into a low-level language-based representation.

48. (Original) A system to translate a language-neutral representation of a compile unit to a low-level language, comprising:

an interface operative to expose methods to control compilation of the language-neutral representation; and

a compiler that implements the interface to translate a plurality of program elements that define the language-neutral representation into the low-level language, each of the plurality of program elements being an instance of a corresponding class of a plurality of classes that represent program constructs, the plurality of elements being arranged according to a hierarchy.

49. (Original) The system of claim 48, wherein the low-level language comprises at least one of an assembly, byte code and intermediate language.